

[illegible]

```

AAAAAA      DDDDDDDDD PPPPPPPP EEEEEEEEEEE RRRRRRRRR RRRRRRRRR 77777777 5555555555 000000
AAAAAA      DDDDDDDDD PPPPPPPP EEEEEEEEEEE RRRRRRRRR RRRRRRRRR 77777777 5555555555 000000
AA         AA DD         DD PP         PP EE         RR         RR RR         RR 77         55         00         00
AA         AA DD         DD PP         PP EE         RR         RR RR         RR 77         55         00         00
AA         AA DD         DD PP         PP EE         RR         RR RR         RR 77         55         00         00
AA         AA DD         DD PP         PP EE         RR         RR RR         RR 77         55         00         00
AA         AA DD         DD PPPPPPPP EEEEEEEEEEE RRRRRRRRR RRRRRRRRR 77         55         00         00
AA         AA DD         DD PPPPPPPP EEEEEEEEEEE RRRRRRRRR RRRRRRRRR 77         55         00         00
AAAAAAAAAA DD         DD PP         EE         RR         RR RR         RR 77         55         0000         00
AAAAAAAAAA DD         DD PP         EE         RR         RR RR         RR 77         55         0000         00
AA         AA DD         DD PP         EE         RR         RR RR         RR 77         55         00         00
AA         AA DD         DD PP         EE         RR         RR RR         RR 77         55         00         00
AA         AA DDDDDDDDD PP         EEEEEEEEEEE RR         RR RR         RR 77         55         000000         00
AA         AA DDDDDDDDD PP         EEEEEEEEEEE RR         RR RR         RR 77         55         000000         00

```

```

LL          IIIIII SSSSSSSS
LL          IIIIII SSSSSSSS
LL          II      SS
LL          II      SS
LL          II      SS
LL          II      SS
LL          II      SSSSSS
LL          II      SSSSSS
LL          II      SS
LL          II      SS
LL          II      SS
LL          II      SS
LLLLLLLLLLL IIIIII SSSSSSSS
LLLLLLLLLLL IIIIII SSSSSSSS

```

AD
Syl
\$S
AD
AD
AD
AD
AD
BU
EM
EX
EX
EX
EX
EX
EX
IS
LO
MA
MA
MA
MC
MC
MM
ND
PR
PR
RE
RE
UA
UB
UB
UB
UN
VA
VE

PS

\$A
WI

Ph
--
In
Co
Pa
Syl
Pa
Syl

(3)	70	EXESUBAERR INT - UBA ERROR INTERRUPT HANDLER
(4)	197	ADAPTER ISR - Adapter power-up interrupt service
(5)	234	LOG ADAPTER - ERROR LOG ADAPTER ERROR
(6)	269	UNMAP ADAPTER ADDRESSES, REPLACE SCB VECTOR
(7)	339	MAP ADAPTER ADDRESS, RESTORE SCB VECTORS

```
0000 1 .TITLE ADPERR750 - I/O ADAPTER ERROR HANDLER
0000 2 .IDENT 'V04-000'
0000 3
0000 4 *****
0000 5
0000 6 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8 * ALL RIGHTS RESERVED.
0000 9
0000 10 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15 * TRANSFERRED.
0000 16
0000 17 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19 * CORPORATION.
0000 20
0000 21 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23
0000 24 *
0000 25 *****
0000 26
0000 27 ++
0000 28 FACILITY: EXECUTIVE, ERROR HANDLING
0000 29
0000 30 ABSTRACT: HANDLE ERROR INTERRUPTS GENERATED BY I/O ADAPTERS ON 11/750.
0000 31 TAKEN FROM ADAPTER ERROR HANDLER FOR 11/780.
0000 32
0000 33 ENVIRONMENT: ENTERED ON INTERRUPT STACK AT ADAPTER IPL.
0000 34
0000 35 AUTHOR: MARYANN HINDEN, Creation date: August 1982
0000 36
0000 37 MODIFIED BY:
0000 38
0000 39 V03-001 KPL0100 Peter Lieberwirth 10-Feb-1984
0000 40 Change to use CONFREG1.
0000 41
0000 42 --
```



```
0000 44 :  
0000 45 : INCLUDED SYMBOL DEFINITIONS  
0000 46 :  
0000 47 $ADPDEF ;DEFINE ADAPTER CONTROL BLOCK SYMBOLS  
0000 48 $EMBDEF <MC,SB,SE> ;DEFINE EMB OFFSETS  
0000 49 $MCHKDEF ;DEFINE RECOVERY BLOCK MASK BITS  
0000 50 $NDTDEF ;DEFINE NEXUS DEVICE TYPES  
0000 51 $PRDEF ;DEFINE PROCESSOR REGISTER NUMBERS  
0000 52 $UASDEF ;DEFINE DW750 IPEC REGISTERS  
0000 53 $VADEF ;DEFINE PFN BITS  
0000 54 :  
0000 55 :  
0000 56 : LOCAL DATA STORAGE  
0000 57 :  
0000 58 : LOCAL MACROS  
0000 59 :  
0000 60 .MACRO VECGEN ; Macro to generate ISR entry point  
0000 61 $$$TEMP=.  
0000 62 PUSHL I^#0 ; Replaced by ADP address for this NEXUS  
0000 63 BRB VECEND ; Join common code  
0000 64 $$$TEMP=-$$$TEMP  
0000 65 ASSUME $$$TEMP EQ 8 ; Each entry must be 8 bytes long  
0000 66 .ENDM  
0000 67  
00000000 68 .PSECT WIONONPAGED,QUAD,RD,WRT
```

```
0000 70      .SBTTL  EXESUBAERR_INT - UBA ERROR INTERRUPT HANDLER
0000 71
0000 72      :++
0000 73      EXESUBAERR_INT - UBA ADAPTOR ERROR INTERRUPT HANDLER
0000 74
0000 75      INPUTS:
0000 76      ON STACK -      PSL
0000 77                      PC
0000 78                      ADDRESS OF ADP + ADP$L_UBASCB+12
0000 79
0000 80      OUTPUTS:
0000 81      NONE
0000 82
0000 83      FUNCTIONAL DESCRIPTION:
0000 84
0000 85      PROCESS ALL INTERRUPTS FROM DW750, SECOND (OPTIONAL)
0000 86      11/750 UBA. LOG AND TRY TO RECOVER.
0000 87      WE GET HERE VIA INTERRUPT DISPATCH CODE IN THE ADP
0000 88      DATA STRUCTURE ITSELF.
0000 89
0000 90      THE FOLLOWING IS A DESCRIPTION OF USEAGE OF ADP DATA
0000 91      STRUCTURE ELEMENTS THAT ARE SPECIFIC TO 11/750 UBA
0000 92      POWERFAIL HANDLING:
0000 93
0000 94      ADP$L_UBASCB      - SAVED ADDRESS OF INTERRUPT SERVICE
0000 95                      CODE IN THE ADP
0000 96
0000 97      ADP$L_UBASCB+4  - INTERRUPT SERVICE CODE:
0000 98                      JSB  @#EXESUBAERR_INT
0000 99
0000 100
0000 101      ADP$L_UBASCB+10 - UNUSED (0)
0000 102
0000 103      ADP$L_UBASCB+12 - SVA OF IPEC REGISTER (CONTAINS POWERFAIL
0000 104                      INTERRUPT ENABLE AND POWER STATUS BITS)
0000 105
0000 106      ADP$L_UBASPT  - ADDRESS OF SPTE THAT MAPS ADAPTER
0000 107
0000 108      ADP$L_UBASPT+4 - ADDRESS OF SPTE THAT MAPS I/O SPACE
0000 109      :--
0000 110
0000 111
0000 112      .ENABL  LSB
0000 113
0000 114      EXESUBAERR_INT::
0000 115      PUSH  #^M<R0,R1,R2,R3,R4,R5>      ; Save GPR's
0000 116      SUBL3  #<ADP$L_UBASCB+10-ADP$L_UBASCB>,SP  ; Find top of ADP block
0000 117      DSBINT  <6*4>(SP),R4                ; Lock out the world
0000 118
0000 119
0000 120      RESTART_UBA:
0000 121      MOVQ  <8*4>(SP),-(SP)                ; Enter here on power up
0000 122      SUBL  #<2*4>,SP                        ; Make second copy of PC,PSL for logger
0000 123      MOVL  ADP$L_UBASCB+12(R4),R3          ; Make room for rest of log
0000 124      SPRTCTINI B*10$,#MCHK$M_NEXM        ; Get address of IPEC register
0000 125
0000 126      ; Protect against machine check

0000004E 3F BB 0000 115      PUSH  #^M<R0,R1,R2,R3,R4,R5>      ; Save GPR's
54 18 AE 8F C3 0002 116      SUBL3  #<ADP$L_UBASCB+10-ADP$L_UBASCB>,SP  ; Find top of ADP block
0008 117      DSBINT  <6*4>(SP),R4                ; Lock out the world
000B 118
0011 119
7E 20 AE 7D 0011 120      RESTART_UBA:
5E 08 C2 0011 121      MOVQ  <8*4>(SP),-(SP)                ; Enter here on power up
53 50 A4 D0 0015 122      SUBL  #<2*4>,SP                        ; Make second copy of PC,PSL for logger
001C 123      MOVL  ADP$L_UBASCB+12(R4),R3          ; Make room for rest of log
001C 124      SPRTCTINI B*10$,#MCHK$M_NEXM        ; Get address of IPEC register
0028 125
0028 126      ; Protect against machine check
```



```
53 63 3C 0028 127      MOVZWL (R3),R3          ; Get contents of register
      002B 128
      002B 129      $PRTCTEND 10$
      002C 130
6E 53 D0 002C 131      MOVL R3,(SP)          ; Save register contents on stack
1F 50 E9 002F 132      BLBC R0,UBA_POWER_DOWN ; No UBA there
      0032 133
      0032 134
      0032 135      : AT THIS POINT IN TIME THE STACK LOOKS AS FOLLOWS
      0032 136
      0032 137      PSL
      0032 138      PC
      0032 139      POINTER INTO ADP or POINTER TO START OF ADP
      0032 140      R5
      0032 141      R4
      0032 142      R3
      0032 143      R2
      0032 144      R1
      0032 145      R0
      0032 146      IPL
      0032 147      PSL (COPY)
      0032 148      PC (COPY)
      0032 149      EMPTY - WILL BE TR # OF THIS ADAPTER
      0032 150      : SP -->IPEC REGISTER
      0032 151
      0032 152
      0032 153      : MAKE SURE THE CORRECT SLOT IS INTERRUPTING AND CORRECT ADP TYPE
      0032 154      : IS IN CONFIG ARRAY
      0032 155
53 50 0C A4 3C 0032 156      MOVZWL ADPSW TR(R4),R0      ; Fetch adapter TR number
      04 AE 50 D0 0036 157      MOVL R0,4(SP)          ; Save on stack
      00000000 GF D0 003A 158      MOVL G^EXESGL_CONFREG,R3 ; Get address of CONFREG array
      53 63 40 D0 0041 159      MOVL (R3)[R0],R3      ; Get adapter type code
      29 53 91 0045 160      CMPB R3,#NDT$_UB1      ; Is adapter UBA1?
      49 12 0048 161      BNEQ 120$                  ; No, inconsistent
      004A 162
      004A 163      : DISPATCH ON STATE OF POWER STATUS BIT IN IPEC REGISTER
      004A 164
      004A 165
6E 2000 8F B3 004A 166      BITW #UASSM_IP_CR1_PDN,(SP) ; Test power status bit
      13 13 004F 167      BEQL UBA_POWER_UP          ; If EQL, power is up
      0051 168
      0051 169      UBA_POWER_DOWN:
      0075 30 0051 170      BSBW UNMAP_UNIBUS          ; Unmap UNIBUS
      0054 171
      0054 172      RETURN:
      005D 30 0054 173      BSBW LOG_ADAPTER          ; Log UBA error
      09 0057 174      .BYTE EMB$K_UBA              ; UBA errorlog type
      5E 10 C0 0058 175      ADDL #<4*4>,SP          ; Clear log from stack
      005B 176      ENBINT                          ; Enable interrupts
      3F BA 005E 177      POPR #^M<R0,R1,R2,R3,R4,R5> ; Restore GPR's
      5E 04 C0 0060 178      ADDL #<1*4>,SP          ; Clear ADP pointer off stack
      02 0063 179      REI                          ; Done
      0064 180
      0064 181      UBA_POWER_UP:                    ; UNIBUS has regained its strength
      3FFE 8F BB 0064 182      POSHR #^M<R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,AP,FP> ; Save regs destroyed
      0068 183      $PRTCTINI B^100$,#MCHK$M_NEXM
```

5C	OC	A4	3C	0074	184	MOVZWL	ADPSW_TR(R4),AP	:	Initialize for this adapter only
		54	DD	0078	185	PUSHL	R4	:	Need ADP after INIT_DEVICE
00000000	'GF	16		007A	186	JSB	G^EXESINIT_DEVICE	:	Initialize devices on this TR level
		52	8ED0	0080	187	POPL	R2	:	Restore register for next routine
00000000	'GF	16		0083	188	JSB	G^UBASINITIAL	:	Re-enable interrupts/init registers
				0089	189	\$PRTCTEND	100\$		
3FFE	8F	BA		008A	190	POPR	#^M<R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,AP,FP>		
	C3	50	E8	008E	191	BLBS	R0,RETURN	:	We made it
	BE	11		0091	192	BRB	UBA_POWER_DOWN	:	No Unibus there
				0093	193				
				0093	194	120\$:	BUG_CHECK UNEXUBAINT,FATAL	:	Crash; wrong/no adapter
				0097	195				


```
0097 197 .SBTTL ADAPTER_ISR - Adapter power-up interrupt service
0097 198 :++
0097 199 : FUNCTIONAL DESCRIPTION:
0097 200 :
0097 201 :     ISR Vector entry point for adapter interrupt service on adapter
0097 202 :     power up after remapping adapter I/O space to black hole page and
0097 203 :     replacing adapter SCB vector pointer. The adapter type is checked;
0097 204 :     Nexus 9 in the configuration array must contain the type for UB1.
0097 205 :
0097 206 : INPUTS:
0097 207 :
0097 208 :     (SP) - ADP address
0097 209 :
0097 210 : OUTPUTS:
0097 211 :
0097 212 :     R0-R5 saved on stack
0097 213 :     R4 <- ADP address
0097 214 :     IPL Raised to 31
0097 215 :--
0097 216 :
0097 217 .ALIGN LONG
0000009A 0098 218 ADP_BASE=+.2 ; Base of ADP address save table
00000098 0098 219 ISR_BASE= ; Base of ISR entry point table
0098 220 VECGEN ; Generate a vector entry point
00A0 221
00A0 222 VECEND:
54 18 3F BB 00A0 223 PUSHR #^M<R0,R1,R2,R3,R4,R5> ; Save GPRs
18 AE D0 00A2 224 MOVL <6*4>(SP),R4 ; R4 <- ADP address
EB 13 00A6 225 BEQL 120$ ; If no ADP address, we're in trouble
00A8 226
00A8 227 DSBINT ; To IPL 31
005C 30 00AE 228 BSBW MAP UNIBUS ; Remap the UNIBUS address space
FF5D 31 00B1 229 BRW RESTART_UBA ; Restart UBA
00B4 230
00B4 231 .DSABL LSB
00B4 232
```



```
00B4 234 .SBTTL LOG_ADAPTER - ERROR LOG ADAPTER ERROR
00B4 235 :++
00B4 236 : LOG_ADAPTER - ERROR LOGGING FOR UBA ADAPTER ERROR INTERRUPTS
00B4 237 :
00B4 238 : FUNCTIONAL DESCRIPTION:
00B4 239 :
00B4 240 : INTERFACE TO ERROR LOGGING ROUTINE IN MACHINE CHECK
00B4 241 :
00B4 242 : INPUTS:
00B4 243 :
00B4 244 : (SP) - Points to error log type code
00B4 245 : (SP)+1 - Return address
00B4 246 : 4 longwords are logged, defined as follows:
00B4 247 : 1 Adapter register (IPEC)
00B4 248 : TR # of this adapter
00B4 249 : PC
00B4 250 : PSL
00B4 251 :
00B4 252 : OUTPUTS:
00B4 253 :
00B4 254 : Entry made in error log
00B4 255 : R0-R3 modified
00B4 256 :--
00B4 257 :
00B4 258 LOG_ADAPTER:
00B4 259
55 30 BB 00B4 260 PUSHR #^M<R4,R5> ; Save R4, R5
OC AE DE 00B6 261 MOVAL <3*4>(SP),R5 ; Address of error log frame
54 10 D0 00BA 262 MOVL #<4*4>,R4 ; Length of error log frame
53 08 BE 9A 00BD 263 MOVZBL @<2*4>(SP),R3 ; Error log type
FF3C' 30 00C1 264 BSBW MCHK$GL_LOG ; Call error logging routine in MCHK750
30 BA 00C4 265 POPR #^M<R4,R5> ; Restore R4, R5
6E D6 00C6 266 INCL (SP) ; Bump Stack past type
05 00C8 267 RSB
```



```
00C9 269 .SBTTL UNMAP ADAPTER ADDRESSES, REPLACE SCB VECTOR
00C9 270 :++
00C9 271 : UNMAP_UNIBUS - Remap UBA address space to black hole page
00C9 272 :
00C9 273 : INPUTS:
00C9 274 :
00C9 275 : R4 - ADP address of the adapter
00C9 276 : IPL 31
00C9 277 :
00C9 278 : OUTPUTS:
00C9 279 :
00C9 280 : Adapter address space (and Unibus I/O space for UBA) mapped to
00C9 281 : black hole page.
00C9 282 : Adapter SCB vector replaced to point to this module
00C9 283 : All general registers preserved.
00C9 284 :
00C9 285 : FUNCTIONAL DESCRIPTION:
00C9 286 :
00C9 287 : On the 11/750, only the second UNIBUS adapter (DW750) may be
00C9 288 : powerfailed; the powerfail interrupt is at 1E4 in the first
00C9 289 : page of the SCB. This routine modifies the SPTE's that map
00C9 290 : the adapter to point to the black hole page instead. The
00C9 291 : SCB vector is modified to point to an entry point in this
00C9 292 : module which will handle the powerup interrupt. Remapping
00C9 293 : to the black hole page allows all code which touches the
00C9 294 : adapter I/O address space to continue without machine checks.
00C9 295 : Note that any such code (drivers, map register initialization,
00C9 296 : etc.) will of course not do what it thought it was doing.
00C9 297 : The result should be device timeout for any I/O request.
00C9 298 :
00C9 299 : The interrupt service routine entry point is located in this module.
00C9 300 : When the adapter is unmapped, the SCB vector is redirected to the
00C9 301 : interrupt service routine in this module. Thus, this module
00C9 302 : receives all interrupts from an unmapped adapter.
00C9 303 :
00C9 304 : The interrupt service routine entry point is of the form:
00C9 305 :
00C9 306 : PUSHL I^#0
00C9 307 : BRB VECEND
00C9 308 :
00C9 309 : When an adapter is unmapped, its ADP address is stored in the immediate
00C9 310 : constant that is pushed on the stack. This location is cleared when
00C9 311 : the adapter is mapped and is used as a flag to prevent double mapping or
00C9 312 : unmapping.
00C9 313 :
00C9 314 : The interrupt service routine is exactly 8 bytes long (a quadword).
00C9 315 : Thus the vector is long word aligned and some assumptions can be made
00C9 316 : for indexing into the array. Indexing must be PIC.
00C9 317 :--
00C9 318 :
00C9 319 UNMAP_UNIBUS:
00C9 320 PUSHR #^M<R0,R1,R2,R3,R6> : Save GPR's used
00C9 321 MOVL #24,R0 : Number of SPTE's to modify
00C9 322 MOVAL W^ADP_BASE,R6 : Get address of ISR
00C9 323 TSTL (R6) : Test saved ADP address cell
00C9 324 BNEQ 100$ : If not zero, already unmapped
00C9 325 MOVL R4,(R6) : Save ADP address for power up
```

004F	8F	BB	00C9	320
50	18	D0	00CD	321
FFC6	CF	DE	00D0	322
	66	D5	00D5	323
	2F	12	00D7	324
66	54	D0	00D9	325


```
51 00000000'GF D0 00DC 326 MOVL G^EXESGL_SCB,R1 ; SCB base address
      56 D7 00E3 327 DECL R6 ; Calculate ISR base +1 (vector) address
01E4 C1 56 D0 00E5 328 MOVL R6,^x1E4(R1) ; Change ISR address for vector in SCB
      52 64 D0 00EA 329 MOVL ADP$SL_CSR(R4),R2 ; Virtual address of adapter space
51 00000000'GF 16 00ED 330 JSB G^MMG$SVAPTECHK ; Address of SPTE that maps adapter
63 00000000'GF D0 00F3 331 MOVL G^EXESGL_BLACKHOLE,R1 ; PFN of black hole page
      15 00 51 F0 00FA 332 10$: INSV R1,#0,#VASS_VPN,(R3) ; Insert new PFN in SPTE
      53 04 C0 00FF 333 ADDL #4,R3 ; Point to next SPTE
      F5 50 F5 0102 334 SOBGTR R0,10$ ; Do for all
      004F 8F BA 0105 335 INVALID ; Invalidate TB
      05 010C 336 100$: POPR #^M<R0,R1,R2,R3,R6> ; Restore GPR's
      RSB
```



```
010D 339 .SBTTL MAP ADAPTER ADDRESS, RESTORE SCB VECTORS
010D 340 :++
010D 341 : MAP_UNIBUS - Map UBA adapter addresses, restore UBA SCB vector
010D 342 :
010D 343 : FUNCTIONAL DESCRIPTION:
010D 344 :
010D 345 : Restore SCB vector address from ADP for the adapter. Restore
010D 346 : SPTE contents that map adapter and I/O address space. These
010D 347 : routines essentially undo everything done by UNMAP_UNIBUS.
010D 348 :
010D 349 : INPUTS:
010D 350 :
010D 351 : R4 - Adapter ADP address
010D 352 : IPL 31
010D 353 :
010D 354 : OUTPUTS:
010D 355 :
010D 356 : PFN field of SPTE's that map adapter and adapter I/O space are
010D 357 : restored to point to the adapter.
010D 358 : SCB vector is pointed to the original system adapter ISR.
010D 359 :--
010D 360 :
010D 361 MAP_UNIBUS:
010D 362 PUSHF #^M<R0,R1,R2,R3> ; Save GPR's
50 44 A4 DE 010F 363 MOVAL ADP$L_UBASCB(R4),R0 ; Address of field containing ISR address
010D 364 BSBB MAP ; Common re-map routine
52 54 A4 D0 0115 365 MOVL ADP$L_UBASPT(R4),R2 ; Saved SPTE value that maps adapter
50 18 D0 0119 366 MOVL #24,R0 ; UBA uses 24 SPTE's
83 52 D0 011C 367 10$: MOVL R2,(R3)+ ; Replace each SPTE
52 52 D6 011F 368 INCL R2 ; Point to next page
50 11 D1 0121 369 CMPL #17,R0 ; Switch from adapter to I/O space
010D 370 BNEQ 20$ ; For Unibus I/O page
52 58 A4 D0 0126 371 MOVL ADP$L_UBASPT+4(R4),R2 ; SPTE value that maps I/O space
EF 50 F5 012A 372 20$: SOBGTR R0,10$ ; All SPTE's
010D 373 INVALID ; Invalidate TB
010D 374 MAP_END:
010D 375 POPF #^M<R0,R1,R2,R3> ; Restore GPR's
010D 376 RSB
```

```
0133 378 :++
0133 379 : MAP - Common adapter re-map routines
0133 380 :
0133 381 : INPUTS:
0133 382 :
0133 383 : R0 - address of ADP field containing saved value for adapter SCB ISR
0133 384 : R4 - address of adapter ADP
0133 385 : return address plus R0,R1,R2,R3 saved on stack
0133 386 :
0133 387 : OUTPUTS:
0133 388 :
0133 389 : R3 - address of first SPTe that maps adapter addresses
0133 390 : Adapter SCB vector restored to system ISR
0133 391 : R0,R1,R2 modified
0133 392 : If Adapter already mapped, stack is cleared and return to caller's
0133 393 : caller is made.
0133 394 :
0133 395 : Refer to comments at start of paragraph 'UNMAP'.
0133 396 :--
0133 397 :
0133 398 MAP:
51 FF63 CF DE 0133 399 MOVAL W^ADP_BASE,R1 ; Get address of vector
61 D5 0138 400 TSTL (R1) ; Adapter already mapped?
18 13 013A 401 BEQL 100$ ; Yes, exit
61 D4 013C 402 CLRL (R1) ; Flag adapter already mapped
51 00000000'GF D0 013E 403 MOVL G^EXE$GL_SCB,R1 ; SCB base address
01E4 C1 60 D0 0145 404 MOVL (R0),^X1E4(R1) ; Restore SCB vector
52 64 D0 014A 405 MOVL ADP$L_CSR(R4),R2 ; Address of Adapter CSR
00000000'GF 16 014D 406 JSB G^MMG$SVAPTECHK ; Address of SPTe that maps adapter CSR
5E 04 C0 0154 407 RSB
D7 11 0157 408 100$: ADDL #4,SP ; Clear callers return address from stack
0159 409 BRB MAP_END ; Restore GPR's and return to caller's caller
0159 410
0159 411 .END
```


ADPERR750
Symbol table

- I/O ADAPTER ERROR HANDLER

H 11

16-SEP-1984 00:47:27
5-SEP-1984 04:06:35

VAX/VMS Macro V04-00
[SYSLOA.SRC]ADPERR750.MAR;1

Page 12
(8)

```
$$$TEMP      = 00000008
ADPSL_CSR    = 00000000
ADPSL_UBASCB = 00000044
ADPSL_UBASPT = 00000054
ADPSW-TR     = 0000000C
ADP_BASE     = 0000009A R    02
BUG$ UNEXUBAINT ***** X    02
EMBSR_UBA    = 00000009
EXESGL_BLACKHOLE ***** X    02
EXESGL_CONFREGL ***** X    02
EXESGL_SCB   ***** X    02
EXESINIT_DEVICE ***** X    02
EXESMCHK-PRICT ***** X    02
EXESUBAERR_INT = 00000000 RG   02
ISR_BASE     = 00000098 R    02
LOG_ADAPTER  = 000000B4 R    02
MAP          = 00000133 R    02
MAP_END      = 00000130 R    02
MAP_UNIBUS   = 0000010D R    02
MCHR$GL_LOG  ***** X    02
MCHK$M_NEXM  = 00000004
MMG$SVAPTECHK ***** X    02
NDT$ UB1     = 00000029
PR$ IPL      = 00000012
PR$ TBIA     = 00000039
RESTART_UBA  = 00000011 R    02
RETURN       = 00000054 R    02
UASSM_IP_CR1_PDN = 00002000
UBASINITIAL ***** X    02
UBA_POWER_DOWN = 00000051 R    02
UBA_POWER_UP  = 00000064 R    02
UNMAP_UNIBUS = 000000C9 R    02
VASS_VPN     = 00000015
VECEND       = 000000A0 R    02
```

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation	PSECT No.	Attributes															
. ABS .	00000000 (0.)	00 (0.)	NOPIC	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE					
\$ABSS	00000000 (0.)	01 (1.)	NOPIC	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE					
WIONONPAGED	00000159 (345.)	02 (2.)	NOPIC	USR	CON	REL	LCL	NOSHR	EXE	RD	WRT	NOVEC	QUAD					

+-----+
! Performance indicators !
+-----+

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.04	00:00:00.67
Command processing	131	00:00:00.36	00:00:01.71
Pass 1	202	00:00:03.23	00:00:11.57
Symbol table sort	0	00:00:00.30	00:00:00.56
Pass 2	91	00:00:00.88	00:00:03.96
Symbol table output	5	00:00:00.03	00:00:00.03

Psect synopsis output	2	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	468	00:00:04.88	00:00:18.52

The working set limit was 1350 pages.

25261 bytes (50 pages) of virtual memory were used to buffer the intermediate code.

There were 20 pages of symbol table space allocated to hold 363 non-local and 8 local symbols.

411 source lines were read in Pass 1, producing 14 object records in Pass 2.

27 pages of virtual memory were used to define 26 macros.

+-----+
! Macro library statistics !
+-----+

Macro library name	Macros defined
-----	-----
\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	17
\$255\$DUA28:[SYSLIB]STARLET.MLB;2	5
TOTALS (all libraries)	22

497 GETS were required to define 22 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:ADPERR750/OBJ=OBJ\$:ADPERR750 MSRC\$:ADPERR750/UPDATE=(ENH\$:ADPERR750)+EXECML\$/LIB

0391 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY